Verifications And Processing Rules

This section covers the following topics:

- Terminology
- General Information
- Using Rules of Verifications in an External Environment
- How Predict Stores Processing Rules
- Generating Processing Rules from Verifications
- Editing the Rule of a Verification
- Changing the Status of a Verification
- Example
- Rippling Verifications

Terminology

Automatic Rule

An automatic rule is used automatically whenever a field to which a verification of status automatic has been linked via *Is verified by VE* is included in a map. Automatic rules cannot be changed in the Natural map editor, which guarantees consistent use of these processing rules throughout an application.

Automatic - Verification Status

A verification has the status automatic if it contains a rule that is linked to at least one field of at least one file for which a DDM has been generated.

Conceptual - Verification Status

A verification of status conceptual contains a rule that has not yet been cataloged with CAT FREE or SA[VE] FREE.

Documented - Verification Status

A verification of status documented does not contain a rule and cannot be used in a Natural map. It is used in the early phases of application design.

Free Rule

A free rule is used in a Natural map by specifying the ID of a verification of status free. The Natural code of the processing rule stored with the verification will be included into the map when the map is cataloged. Free rules can be defined and modified with the Natural map editor and directly in Predict.

Free - Verification Status

A verification has the status free if it contains a rule that has been cataloged with CAT FREE or SA[VE] FREE.

Inline Rule

Inline processing rules are defined within a Natural map source and do not have a name assigned. Inline rules in Natural can be used independently of Predict.

Natural Construct - Verification Status

A verification of status Natural Construct is created by entering command CAT N or SA[VE] N in the Rule Editor. These verifications are only used by Natural Construct.

Processing Rule

Rule for validating data entry in a Natural map or SQL database. The following types of processing rules can be defined:

For Natural:

- Inline processing rules
- Free rules
- Automatic rules

For SQL:

- SQL

Rule Editor

The Rule Editor is a modified Natural Editor in Predict which is used to edit the rule of a verification. See the section Editors in Predict in the **Predict Reference documentation**.

SQL - Verification Status

A verification of status SQL is created by entering command CAT S or SA[VE] S in the Rule Editor. See Verifications of Status SQL.

Verification

A verification is a predefined object type in Predict and can contain the Natural or SQL code of a processing rule.

General Information

Natural processing rules perform validity checks on input data to ensure that the data to be processed is suitable. For example, in a program controlling traffic lights, the only input values allowed for the field colour might be red, green and amber.

Natural processing rules can be defined and stored centrally in Predict.

Benefits

Storing processing rules in Predict has the following advantages:

- Programming costs and the number of errors can be reduced by using free rules stored in Predict.
- The use of processing rules can be forced by linking automatic rules to fields via association *Is verified by VE*. Automatic rules cannot be changed in the Natural map editor. This guarantees consistent use of these processing rules throughout an application.
- The use and design of processing rules can be planned and revised by using verifications of status documented and conceptual.

Rules Applying to Processing Rules in Predict

The following general rules apply to processing rules in Predict:

- Processing rules of status documented or conceptual can be linked to fields via *Is verified by VE*.
- Natural inline processing rules can be integrated into Predict as free rules by giving them a name in the Natural map editor.
- Automatic rules can be used as free rules by specifying their name in the map.
- Processing rules can be written in either structured mode or report mode.
- A GENERATE command creates the Natural code of a processing rule from the rule of a verification. The generated code can then be changed to meet specific requirements.
- Links from fields to verifications linked via *Is verified by VE* are rippled. See Rippling Verifications

Using Rules of Verifications in an External Environment

The status of the Predict verification object determines how a rule stored with the verification is used in an external environment. Six status types are distinguished:

- Documented
- Conceptual
- Free
- Automatic
- Natural Construct
- SQL

The characteristics of the different status types are described below.

Verifications of Status Documented

Verifications of status documented can be used in the early phases of application design when the parts of an application that have to be implemented are listed. verifications of status documented do not contain processing rules and cannot be used in Natural maps.

Documented	
ID:	xyz
Type:	equal
Format:	alpha
Value	’xyz’

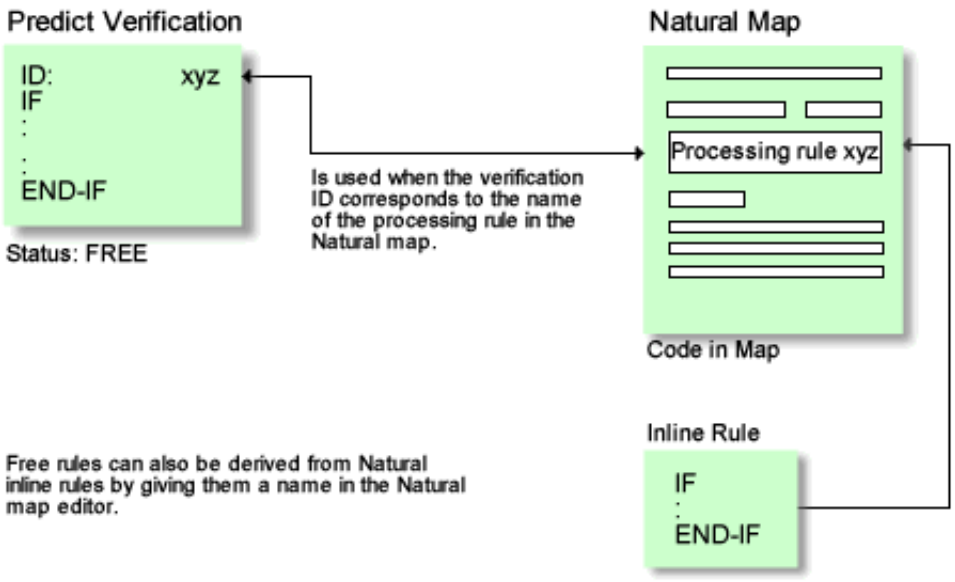
Verifications of type documented are not connected to an external environment.

Verifications of Status Conceptual

Verifications containing a processing rule that has not yet been cataloged with CAT FREE or SA[VE] FREE have the status conceptual.

Verifications of Status Free

A verification has the status free if it contains a processing rule that has been cataloged with CAT FREE or SA[VE] FREE and can therefore be used in any Natural map by linking it explicitly to a field.



A free rule is used in a Natural map by specifying the verification ID. A Select function is provided for selecting free rules from Predict. Only free rules with a format compatible with the format of the input field of the map for which they are to be used will be displayed by the select function. The Natural code of the processing rules stored under the given verification ID will be included into the map when the map is cataloged.

Free rules can be defined and modified with the Natural map editor and directly in Predict.

Verifications of Status Automatic

A verification has the status automatic if it contains a processing rule that is linked to at least one field of at least one file for which a DDM has been generated. An automatic rule is automatically used every time a field to which it has been linked is included in a map. Automatic rules are centrally defined by the administrator who generates DDMs and cannot be modified by individual programmers with the Natural map editor. Defining an automatic rule is a two-stage process:

1. Link the verification containing the rule to a field of a file (a real file or a userview) in Predict.
2. Activate the rule by generating a DDM for that file.

Rules Applying to Automatic Rules

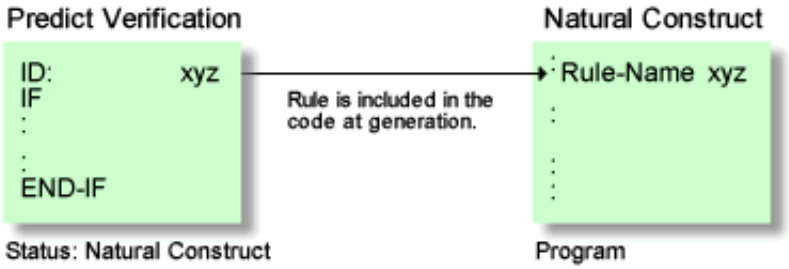
The following rules apply for the use of automatic rules:

- Up to 50 automatic rules can be linked to a field.
- If the code contained in an automatic rule is changed but its links to fields remain, the Predict Replace Verification Rule function can be used to update the active code which is used by the fields. There is no need to regenerate the DDM. Natural maps using processing rules that have been modified should be recataloged to ensure that they use the new version of the processing rule.
- The LIST XREF function with the option Save set set to Y can be used to recatalog maps efficiently.
- An automatic rule can also be used as a free rule by specifying the name of the corresponding verification in a Natural map. Automatic rules that have been used in this fashion cannot be modified with the Natural map editor.
- Automatic rules cannot be changed with the Natural map editor.

Verifications of Status Natural Construct

Verifications of status Natural Construct can be accessed only from Natural Construct.

These verifications must be linked to a field so that Natural Construct can access them. Unlike verifications of status Automatic, it is not necessary to generate a DDM for a verification of status Natural Construct in order that the rule is used.



Verifications of Status SQL

The following rules apply:

- Verifications of status SQL are skipped by functions Generate DDM and Replace Verification rule.
- The syntax of the processing rules is not checked by Predict.
- These processing rules may only contain references to the field to which they are linked. Ampersand notation is used instead of a fixed allocation.
- The ampersand references are replaced by the corresponding field name when the CREATE TABLE statement is generated.
- If you execute the command GEN[ERATE] S in the Rule Editor, a corresponding SQL clause is created for all rule types except user routine. When the code of the rule is saved, the status of the verification is changed to SQL.
- Only one verification of status SQL (and 49 of a status other than SQL) may be linked to a field. This condition is checked only when the CREATE TABLE statement is generated.
- Comments are removed when the CREATE TABLE statement is generated.
- Otherwise the handling of these verifications corresponds to verifications of status Natural Construct. See Verifications of Status Natural Construct.

How Predict Stores Processing Rules

Processing rules are stored in Predict as attributes of verifications. If a verification has been linked to a field of a map, the Natural code of the rule is inserted when that map is cataloged.

Variable Names in Processing Rules

In the source code of a processing rule, the name of a variable can be represented by an ampersand (&). The Natural compiler or Predict generation function will substitute the name of the field (or PF key for a PF key rule) for the ampersand. This allows the use of a rule for different fields.

Example:

IF & = ’ ’ REINPUT ’ENTER NAME’ MARK *&

Priority of Processing Rules

1. Processing rules assigned to function keys have highest priority.
2. Rules linked to different fields of a map are executed in the order in which the fields appear on a terminal screen.
3. A rank from 0 to 99 can be allocated to each inline rule or free rule linked to a field of that map. Additionally a rank can be allocated to all automatic rules linked to a field.
   The rules linked to that field will then be executed in ascending order of rank.
4. Automatic rules linked to a field in Predict are executed in the order their Predict verification IDs appear in the verification list of the field.

Processing Rules and Field Formats

Every rule is allocated a format to ensure that the rule will be compatible with the field format. The following table lists the compatible combinations of field format and rule format.

Field Format	Compatible Rule Format
A	A,B
N,P	N
I,F	B
B =< 4	A,B,N
B > 4	A,B
D (date/time)	D (date/time)
L	L

The rule format K (function key) applies exclusively to free rules.

Generating Processing Rules from Verifications

Processing rules can be created from Predict objects of type verification. Follow the steps below:

1. Call the Rule Editor by entering Y in the field labelled EDIT Rule in an Add a Verification, Copy Verification or Modify Verification screen or by executing the Edit rule of a Verification function (code R).
2. Enter the GENERATE [S|N] command in the Rule Editor to generate a first version of the processing rule from the definitions in the verification.
3. Modify the processing rule as required.
4. Test the rule with the RUN or CHECK command (Natural rules only).
5. The rule is cataloged/saved with either one of the following commands:
   - SA[VE] [[FREE] RET[URN]]
   - CAT [[FREE] RET[URN]]
   - SA[VE] [S|N]
   - CAT [S|N]

If FREE is used, the rule is stored as a free rule.

Note:
Commands SAVE or CAT do not perform a syntax check. The syntax is checked when you catalog a map that uses the rule or when the CREATE TABLE statement is executed.

Editing the Rule of a Verification

The rule of a verification is edited with the Predict Rule Editor. This editor can be invoked in one of the following ways:

- Enter Y in the field EDIT Rule in the bottom line of the Add a Verification, Copy Verification or Modify Verification screen.
- Call function Edit rule in the Verification Maintenance Menu.
- Enter command EDIT VERIFICATION RULE <Verification-ID>

Editor Commands

Note:
This section describes rule-specific editor commands. General editor commands are described in the section Editors in Predict in the **Predict Reference documentation**.

CAT [[FREE] RET[URN]], SA[VE] [[FREE] RET[URN]]	Catalog/save the edited rule as a free rule. This command is only available when creating new rules and when editing conceptual rules. **Note:** The commands SAVE and CAT do not perform a syntax check. The syntax is checked however, when you catalog a map that uses a rule.
C[HECK]	Checks whether the edited rule's Natural syntax is valid and reports errors.
GEN[ERATE]	Generates a processing rule from the values defined in the rule of the verification and adds it to the end of the Natural source in the rule editor. This command is not available for verifications of type User routine. A table which shows the Natural and SQL statements generated for the different verification types can be found in the section Rule Editor in the **Predict Reference** documentation.
GEN[ERATE] N	Generates a rule for Natural Construct from a verification of status documented (D). The status of the verification is changed to N.
GEN[ERATE] S	Generates an SQL clause for all verification types except user routine. When the code is saved, the status of the verification is changed to S.
GLOBALS SM=OFF	Switch to the reporting mode of Natural.
GLOBALS SM=ON	Switch to the structured mode of Natural.
RENUM[BER], N	Renumber the source lines in steps of N and renumber references to them accordingly.

<table>
<tr>
<td>RUN</td>
<td>Checks the edited rule. If no errors are found, a map is produced with which the user can test the rule by entering input values. The following rules apply:

Length and format of the input field are derived from the rule format. For rules with format A, B or N, an additional window is displayed, where the derived field length can be overwritten.

<table>
<tr><th>Rule Format</th><th>Format of the derived field</th><th>Length of the derived field</th></tr>
<tr><td>A</td><td>A</td><td>66</td></tr>
<tr><td>B</td><td>B</td><td>33</td></tr>
<tr><td>D</td><td>D</td><td></td></tr>
<tr><td>L</td><td>L</td><td>1</td></tr>
<tr><td>N</td><td>N</td><td>27</td></tr>
</table>

RUN tests a rule of format K (function key) without input data.
For a rule of format L (logical), a blank space means false and any other input value means true.
The stack must not be changed.
The contents of the source area must not be changed.

Note:
All variables used except the ampersand (&) must be defined within the code.

The variable names SYSDIC-C1 and SYSDIC-C2 are used for internal purposes and must not be used within the rule.
The source will be renumbered.

</td>
</tr>
</table>

Changing the Status of a Verification

Predict assigns the status of verifications itself. The following table shows which actions cause a change of status.

Old Status	New Status	Action
inline	free	Give the rule a name in the map editor.
documented	conceptual	Add a rule to the verification.
conceptual	free	Either catalog the rule in the rule editor with the command SAVE FREE or CAT FREE or use the Rename Verification function to change the status explicitly.
conceptual	SQL, Natural Construct	Catalog the rule in the rule editor with the command SAVE S|N or CAT S|N. At least one line must have been changed before cataloging.
free	inline	Change the rule's name to a blank in the map editor. The rule will still exist in Predict with status free.
conceptual	automatic	Link the rule to at least one field (with the field maintenance function Link Verification), then generate a DDM for the file which includes it.
free	automatic	Link the rule to at least one field (with the field maintenance function Link verification), then generate a DDM for the file which includes it.
free	conceptual, Natural Construct	Use the Rename Verification function. The status of a free rule cannot be changed to conceptual if the rule is used in any Natural map.
automatic	conceptual	Unlink all fields from the rule (with the field maintenance function Link Verification) then regenerate the related DDMs. If the rule is also used as a free rule, the status of the verification will be changed to free.
documented	SQL, Natural Construct	Generate a rule for Natural Construct from a verification of status documented (D) using the GEN[ERATE] N command of the Rule Editor; for SQL using the command GEN[ERATE] S.
Natural Construct	free, conceptual	Use the Rename Verification function.

Example

One of six town names are allowed as input. The verification describing this validity check is created with values as shown below:


```
10:13:40            *****  P R E D I C T  4.2.2  *****              2002-07-31
                         - Modify Verification -
Verification ID . TEST-TOWN                          Modified 2002-07-31 at 09:46
Status .......... Free                                     by HNO
Keys ..                                                              Zoom: N

Format .........* A  Alphanumeric                          Modifier    Zoom: N
Type ...........* T  Table of values
Message nr ......
Replacement 1 ...
Replacement 2 ...
Replacement 3 ...
Message text .... No SAG-office in that town.

Abstract     Zoom: N             Values     Zoom: N
                                   BRUESSEL
                                   RESTON
                                   PARIS
                                   DERBY
                                   CAMBRIDGE
                                   DARMSTADT

EDIT:   Owner: N   Desc: N * Rule: N
```

The following processing rule is generated if the GENERATE command in the Rule Editor is applied to this verification.

```
* ************************************************************************
* Verification: TEST-TOWN generated by PREDICT                           *
*  with format: Alphanumeric; Type: Table of values;                     *
*           on: 2002-07-31; at: 10:13:33; from user: HNO                 *
* ************************************************************************
IF NOT ( & = 'BRUESSEL' OR = 'RESTON' OR = 'PARIS' OR = 'DERBY'
OR = 'CAMBRIDGE' OR = 'DARMSTADT' )
        REINPUT 'No SAG-office in that town.'
           MARK *&
```

Rippling Verifications

Rippling Verifications from Standard Files

Each field of a standard file can have a list of verifications via association *Is verified by VE*, which apply to that field. When the list is edited, corresponding changes are automatically made in the verification list of every field related to that standard field, according to the following rules:

- Every verification contained in the verification list of a standard field must also be contained in the verification list of a field related to that standard field. However, the sequence of verifications in the lists can differ.
- If a verification ID is changed, the same change is automatically made to that verification ID everywhere it appears in a verification list of related fields.
- If a verification ID is deleted, every instance of that verification ID is automatically deleted from the verification list of every related field.
- If a verification ID is added anywhere in the list, the same verification ID is automatically added to the end of the verification list of every related field.
- A verification ID can be removed from verification lists of related fields that are marked as no check against standard.

Rippling Verifications from Physical Files to Userviews

Fields of physical files can have verifications linked to them via *Is verified by VE*. When a list of verifications linked to a field in a physical file is modified, corresponding changes are automatically made in the verification list of userview fields related to that field in the file. The following rules apply:

- The verification list of a field in a userview does not have to contain all the verifications that are contained in the list of the physical file field from which the userview field has been related.
  Hence, verifications can be deleted from the verification lists of userview fields, after these have been related to physical files.
- If a verification ID is changed, the verification ID is changed in the verification lists of all related fields.
- If a verification ID is deleted, every instance of that verification ID is automatically deleted from the verification list of every related field.
- If a verification ID is added, it is added to the verification lists of related fields.

